Aalto University
School of Science

Machine Learning Systems in Edge-Cloud Continuum

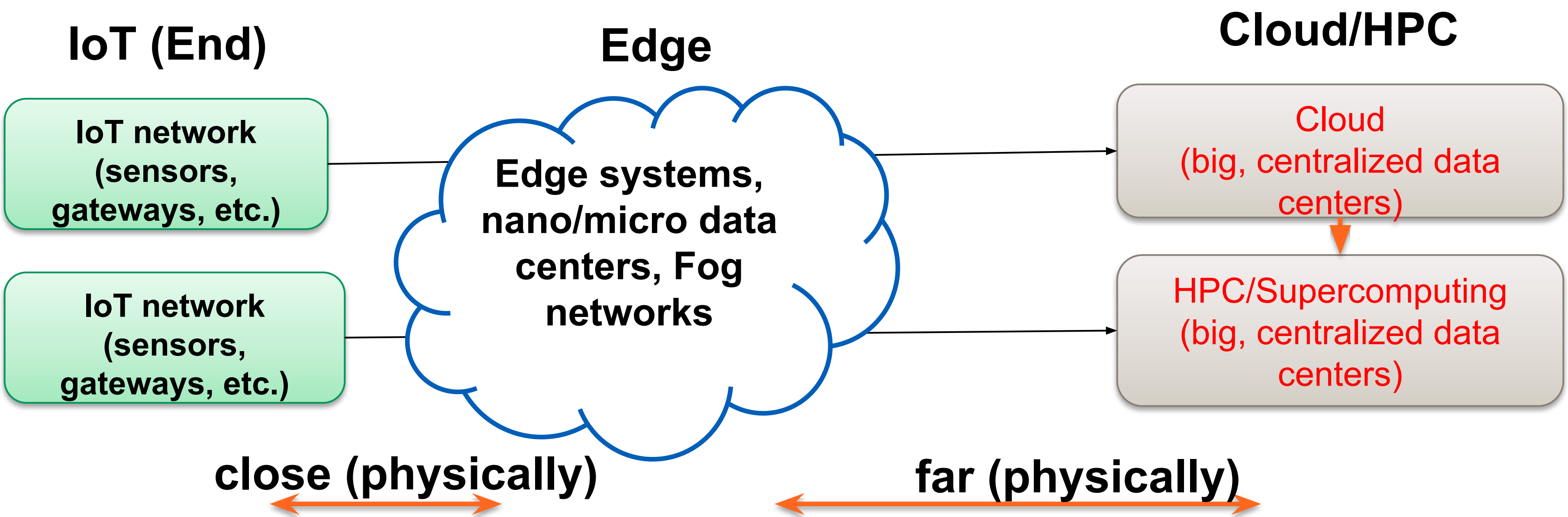
Hong-Linh Truong
Department of Computer Science
linh.truong@aalto.fi, https://rdsea.github.io

CS-E4660 Advanced Topics in Software Systems, Fall 2023
4/10/2023

Learning objectives

- **Understand and analyze the relationship between edge computing and ML systems**
- **Explore and study basic concepts and issues when engineering ML systems in edge-cloud continuum**
- **Identify and work on ML system optimization problems across levels of abstraction in edge-cloud continuum**

IoT-Edge-Cloud



"Edge" is just an abstraction view ("near edge" vs "far edge")

Edge computing

- **Distributed computing at the edge and end devices**
  - *many distributed low-end as well as a limited number of high-end devices/machines for different purposes*
- **Leveraging common technologies like in the cloud and specific ones**
  - *e.g., virtualization, container orchestration, messaging systems, storage/database, Web services*
- **But with different constraints**

Edge computing

- **Computation/analytics can be done at the edge**
  - where data is generated/collected, close to the data sources
    - *next to IoT devices and sensing equipment*
  - many distributed (moving) locations, e.g., in the shopping center, in the car
- **Near real-time data processing and analytics needed in most situations**
- **High heterogeneity w.r.t system models, hardware architectures, network connectivities, and protocols**

Machine learning in the edge

- **Many applications can benefit from ML/data analytics capabilities**
  - inferencing/classification in mobile devices/smart homes, healthcare
  - (near) real-time ML-based steering (autonomous cars, speech control, traffic controls)
  - (near) real-time anomaly detection and forecasting: fraud detection, fault detection, and incidents (e.g., in manufacturing, network operations, security monitoring)
  - safety and quality control

Machine learning/big data analytics in the edge

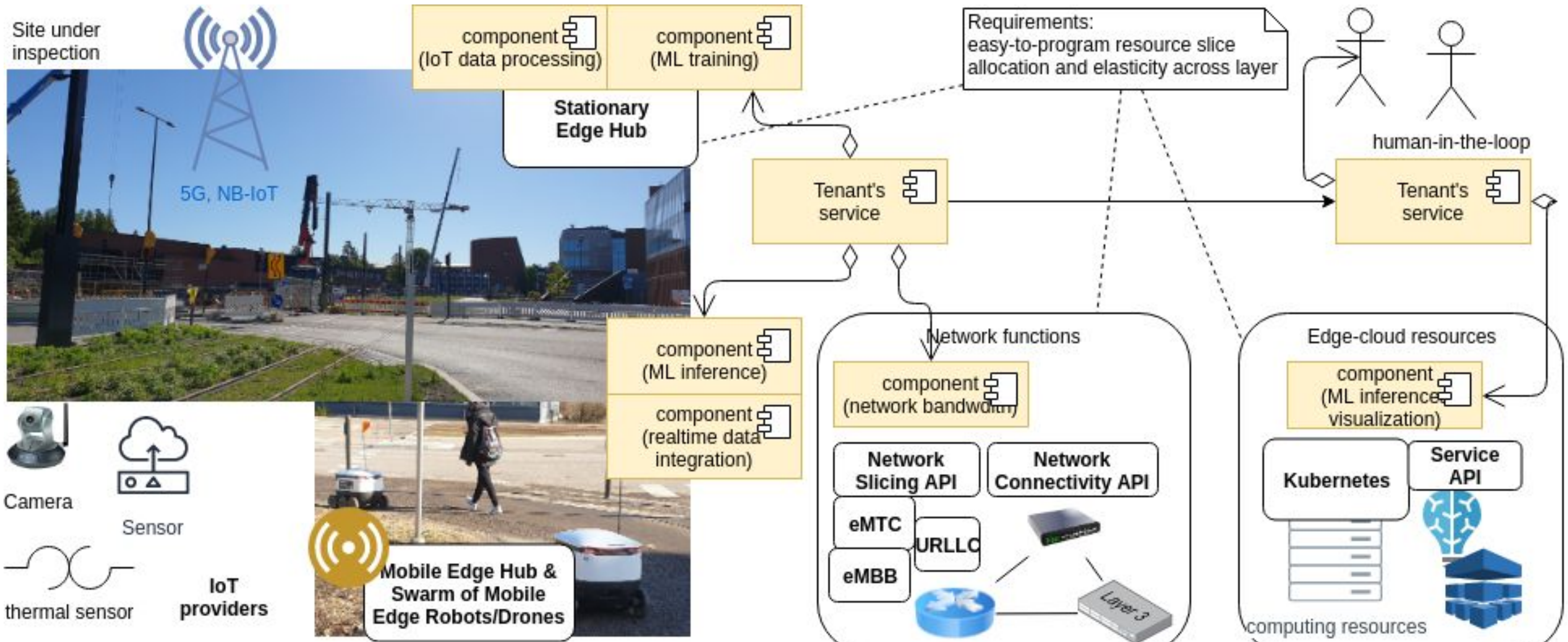
- **Close to data sources □ "data locality" benefits**
  - security & privacy
  - performance (low latency, real-time response)
- **Customization/personalization**
  - deployment diversity, cost saving
  - working under no connectivity
- **But with many challenges!**

Edge-cloud continuum

- **Enable widely decentralized complex workloads (AI/ML, data analytics, real-time operations, etc)**
  - in distributed and heterogeneous environments
- **Interconnecting edge and cloud resources and using them in similar manners**
  - like in the same system to, moving tasks seamlessly between edge and clouds
  - reduce issues and costs due to diverse development, deployment and operations
- **From the technical view point**
  - use similar enabling technologies, like containers and orchestrators
  - employ similar management techniques and methods (deployment, monitoring, etc.)

Example of a scenario

Training/ML inferences: move from the cloud to the edge



In city blocks, villages, etc.

Source: https://www.researchgate.net/publication/361261810_Robustness_via_Elasticity_Accelerators_for_the_IoT-Edge-Cloud_Continuum

Aalto University
School of Science

Basic concepts/issues when engineering ML in edge systems

CS-E4660 Advanced Topics in Software Systems, Fall 2023
4/10/2023

Things affecting robustness, reliability, resilience and elasticity

- **Network problems**
  - high latency, low-bandwidth, unreliable connectivities
- **Computation capabilities**
  - constrained processing power, a lot of specific chips and accelerators, and limited memory
- **Storage is not enough for big data (as sources)**
- **Energy/power usage of devices/machines**
- **V* issues in data**
  - out of distribution data, unlabeled data, time series data, streaming data

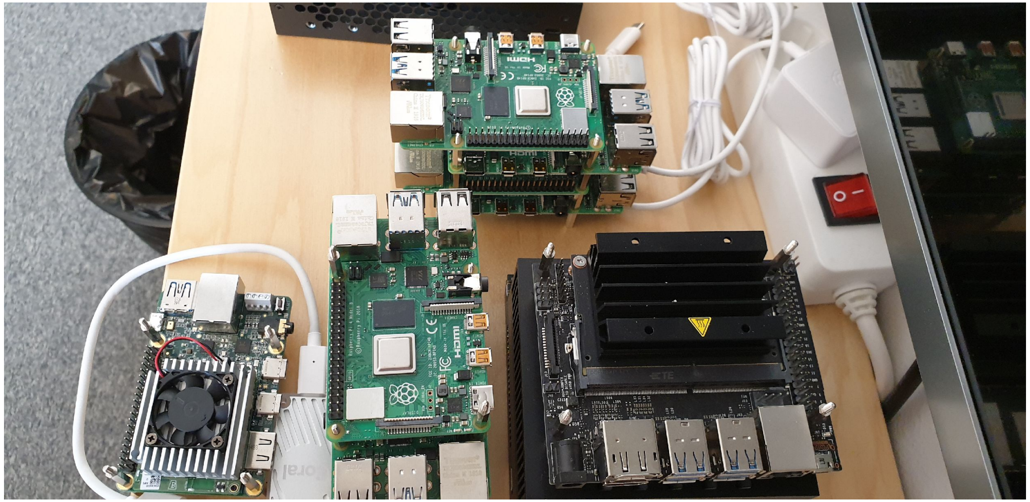
Things affecting ML capabilities

- **Edge with hardware heterogeneity**
  - common hardware (e.g., AMD, Intel, ARM), SoC and microcomputers, microcontrollers
  - with/without common and AI-based accelerators like FPGA, GPU, and TPU

⇒ requirements for certain types of ML might not be fulfilled: computation-intensive ML

Pervasive embedded edge devices

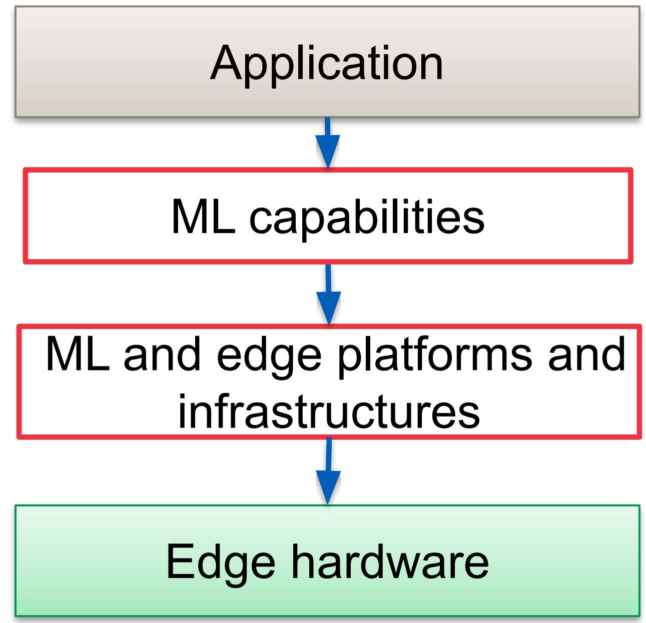
- Raspberry PI4
- Google Coral
- Jetson Nano
- Xilinx
- A huge number of MCUs (Microcontroller Units) → TinyML



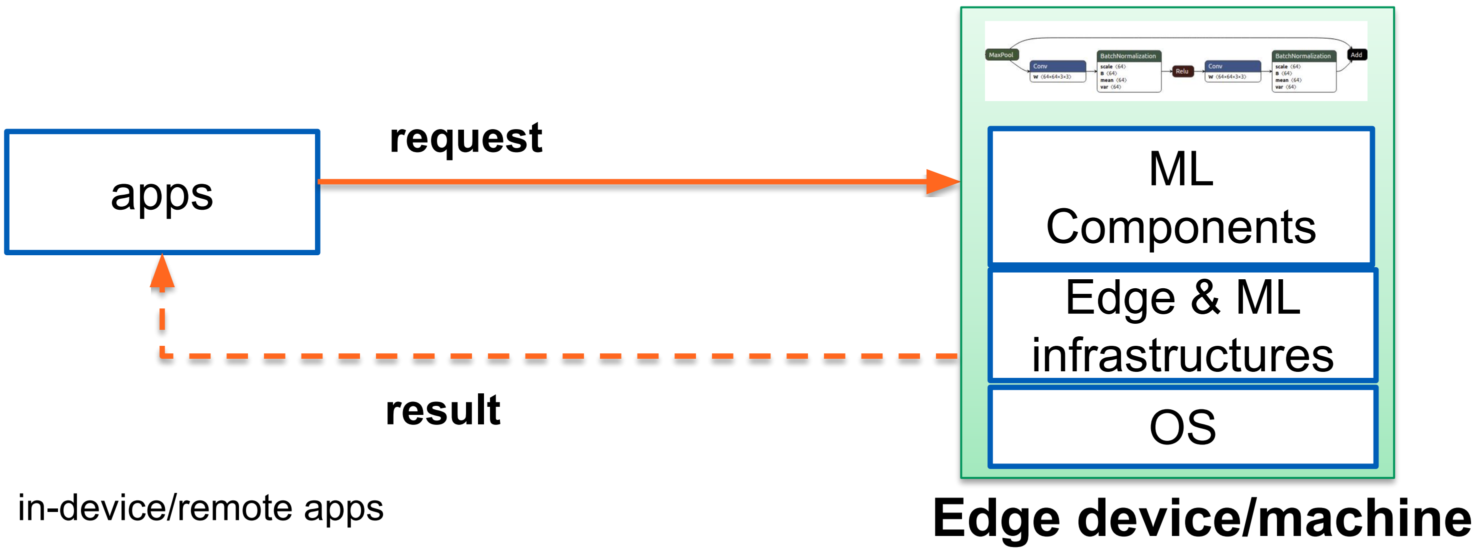
→ challenges with software libraries, updates, etc.

Interaction models in edge-cloud ML systems

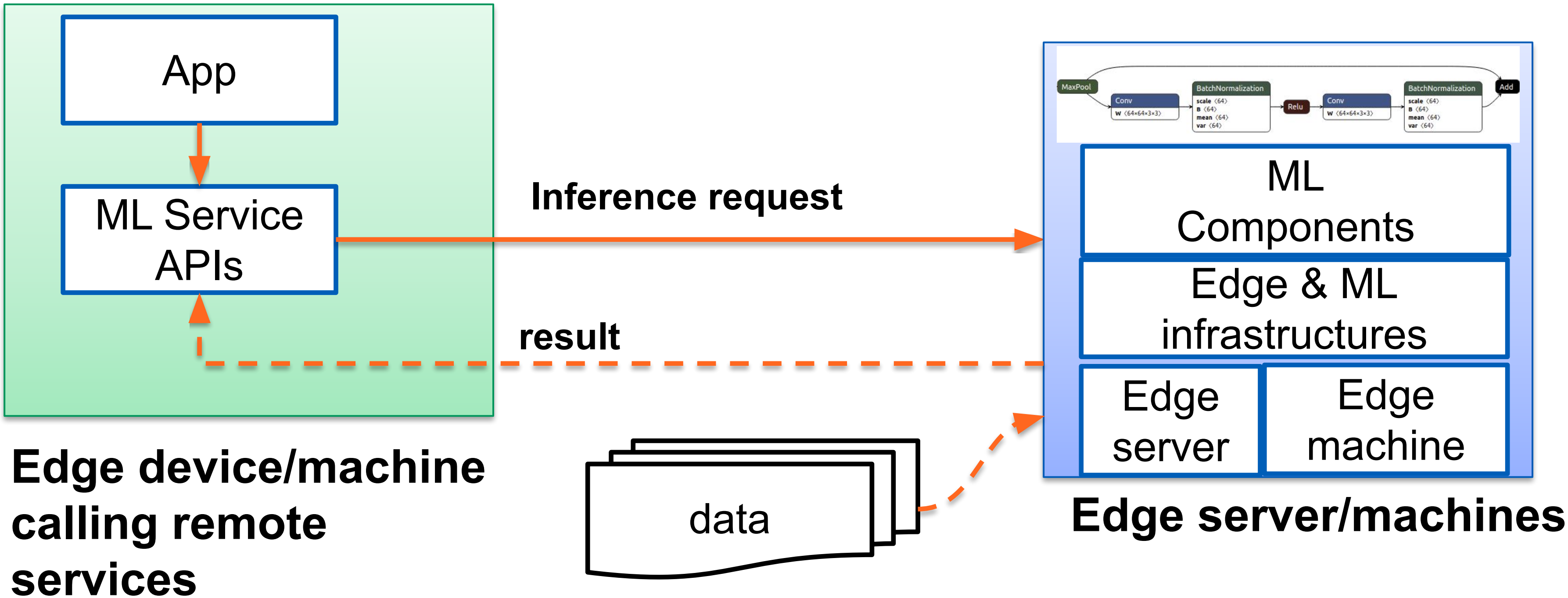
Which components do what, and where are they?



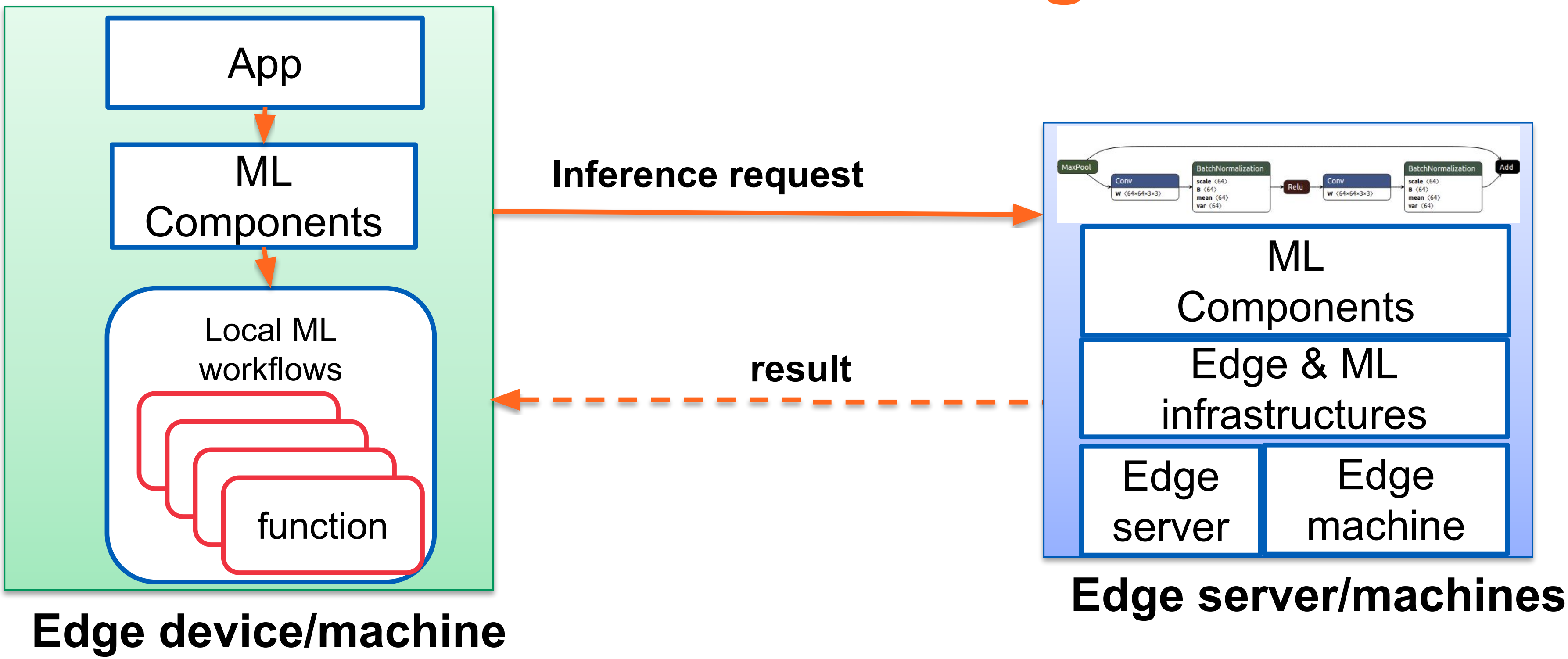
Interaction models: Stand-alone/in-device ML capabilities



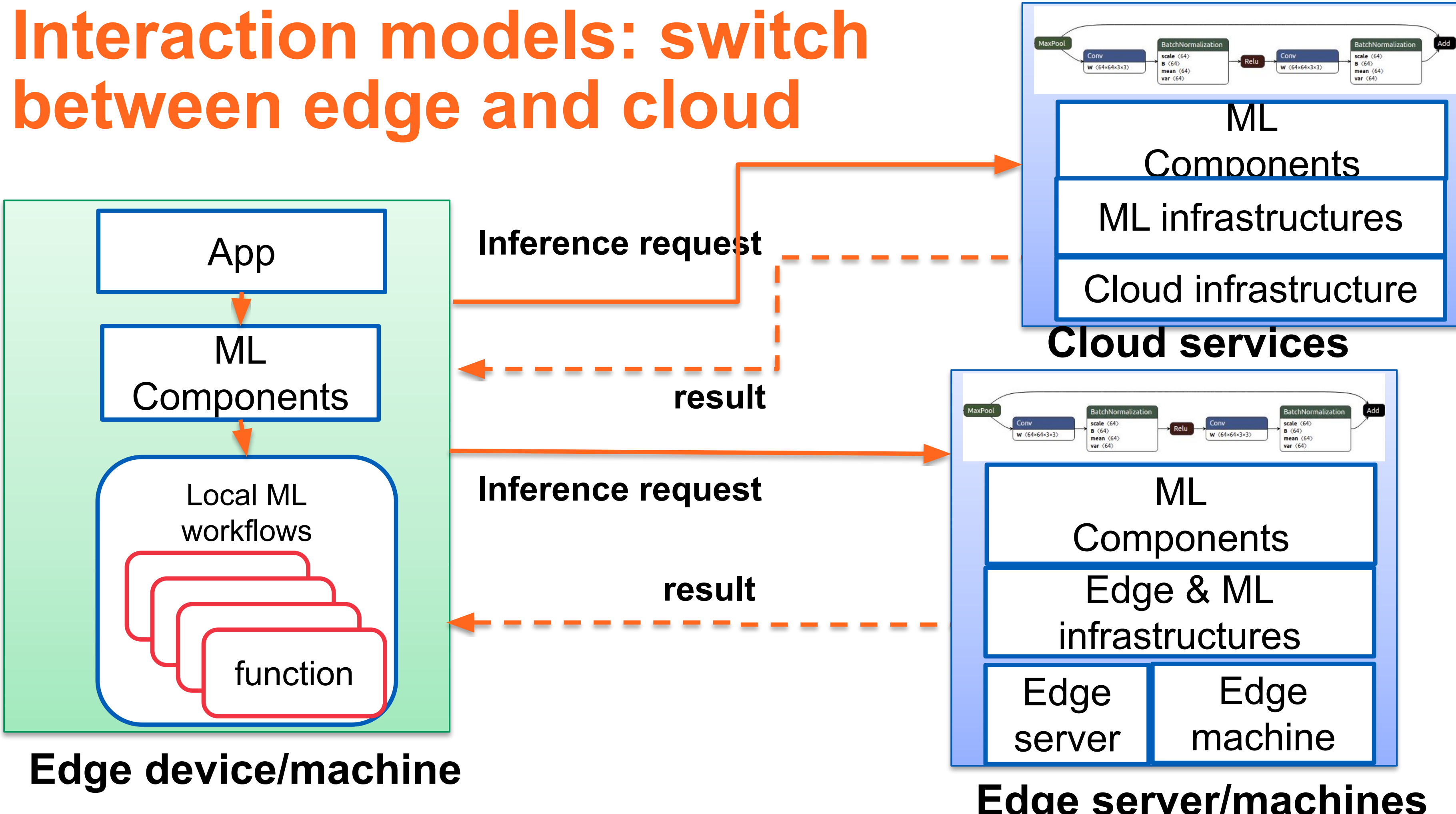
Interaction models: common client-server



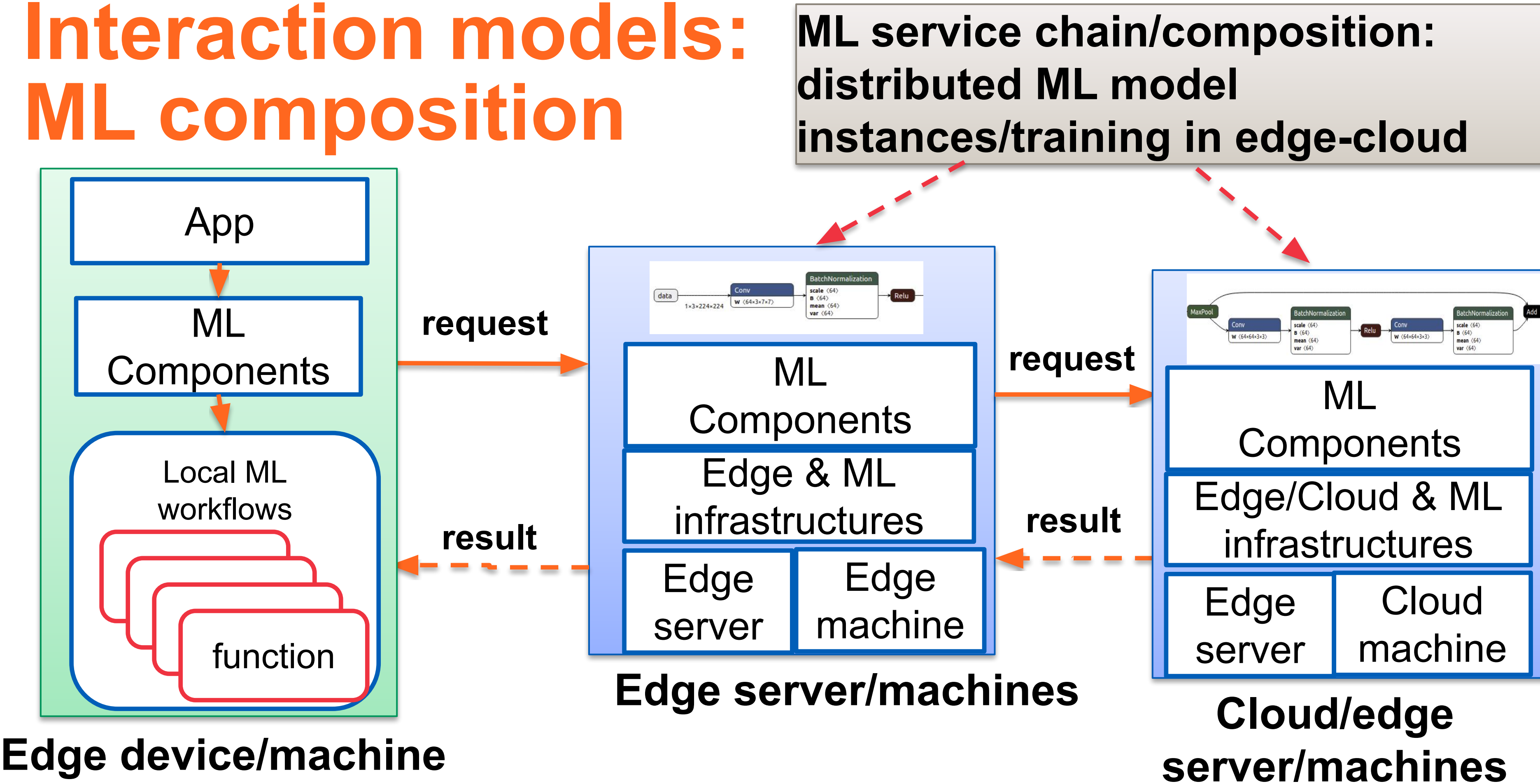
Interaction models: pre-processing and remote ML offloading



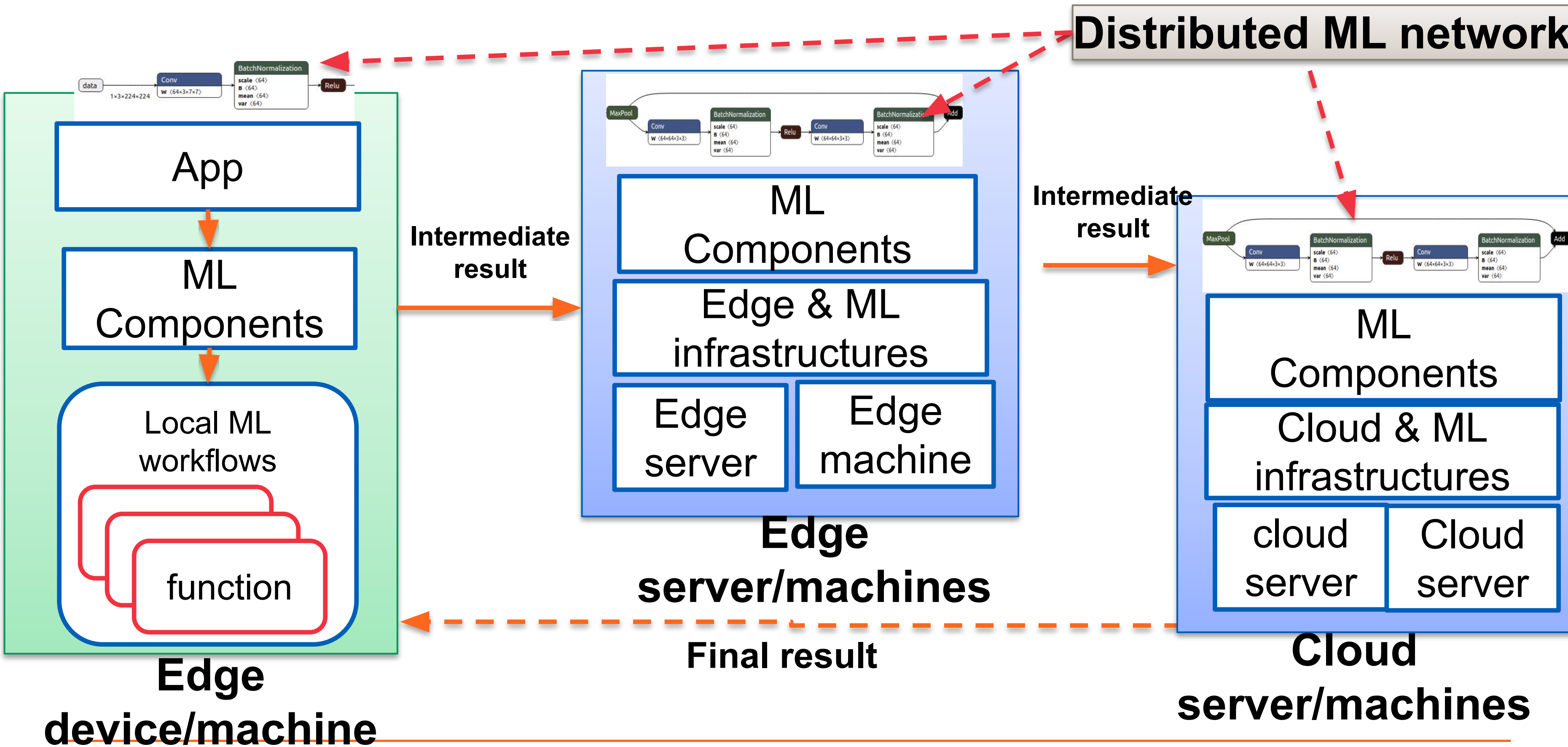
Interaction models: switch between edge and cloud



Interaction models: ML composition



Interaction models: ML composition

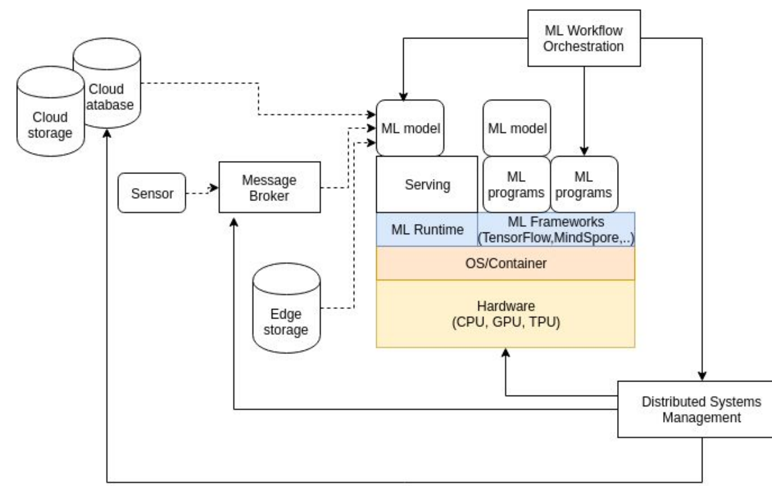


Requirements for these interactions

- **Which application needs the inference and for which purposes?**
  - performance
  - ML inference accuracy
  - data privacy
  - security
- **e.g. in Manufacturing**
  - quality control → control the manufacturing process → very fast
  - some quality control requires very accuracy

Example of key requirements for suitable ML and runtime in edge nodes

- **Energy consumption**
- **Resource constraints**
  - less computation capabilities □ precision and accuracy?
- **Latency and uncertainty**
- **Interfaces with different networks capabilities**
- **Support accelerators**
  - e.g., FPGA, AI Accelerators (e.g. Intel® Movidius Myriad X VPU)
- **Trade-offs between generic versus specific features**



Some ML frameworks and runtime for the edge

- **TF-lite (https://www.tensorflow.org/lite)**
- **https://github.com/Microsoft/EdgeML**
- **uTensor: https://github.com/uTensor/uTensor**
- **Android NN (https://developer.android.com/ndk/guides/neuralnetworks)**
- **CoreML (https://developer.apple.com/machine-learning/core-ml/)**
- **PyTorch mobile (https://pytorch.org/mobile/home/)**
- **Snapdragon Neural Processing Engine SDK**
  - https://developer.qualcomm.com/docs/snpe/overview.html

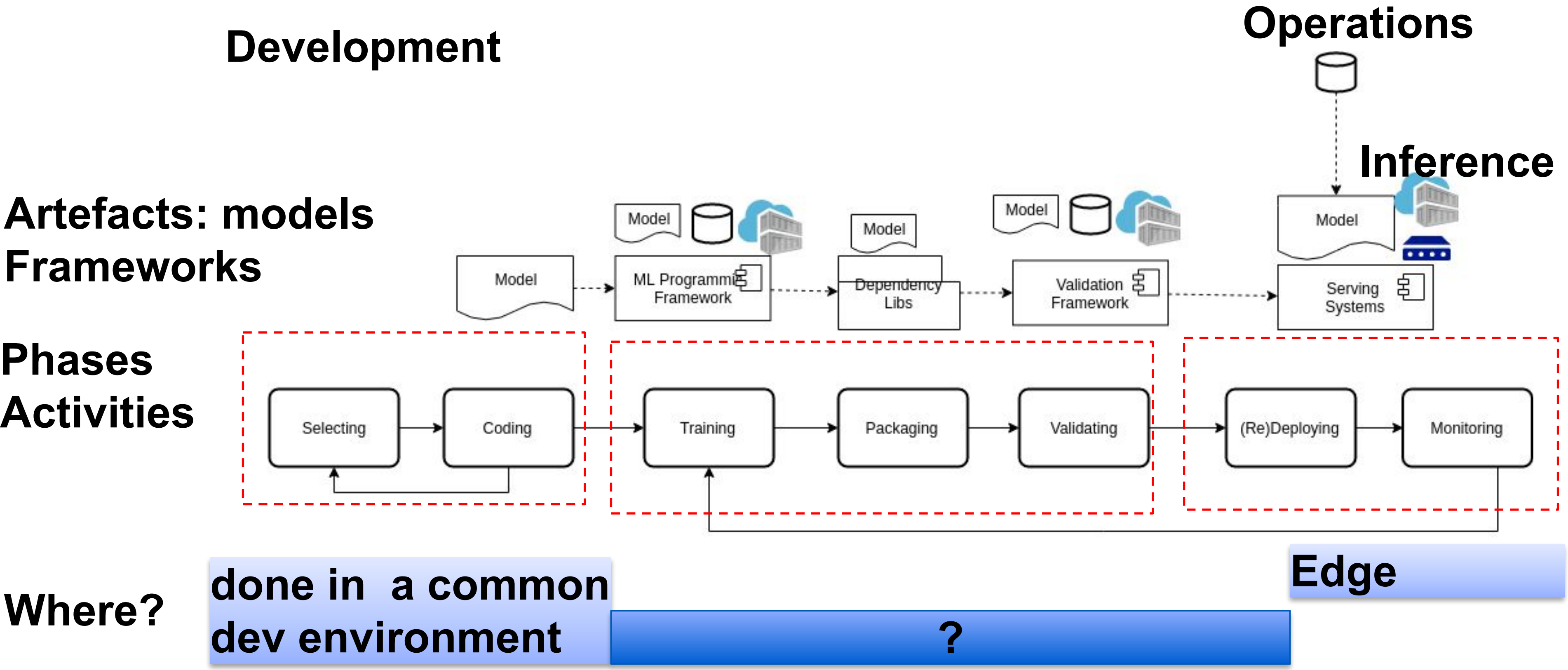
Changes in MLOps

- **MLOps (ML DevOps)**
  - DevOps principles for ML
  - in ML engineering processes: key artefacts are ML models, data and runtime libs
- **Changes in ML with edge systems**
  - DevOps and DataOps activities in the edge
  - optimization and training activities
  - testing and benchmarks
  - monitoring

Check:
https://cloud.google.com/solutions/machine-learning/mlops-continuous-delivery-and-automation-pipelines-in-machine-learning

https://aws.amazon.com/blogs/machine-learning/demystifying-machine-learning-at-the-edge-through-real-use-cases/

MLOps in edge systems



Train in clouds/on-premise but edge deployment

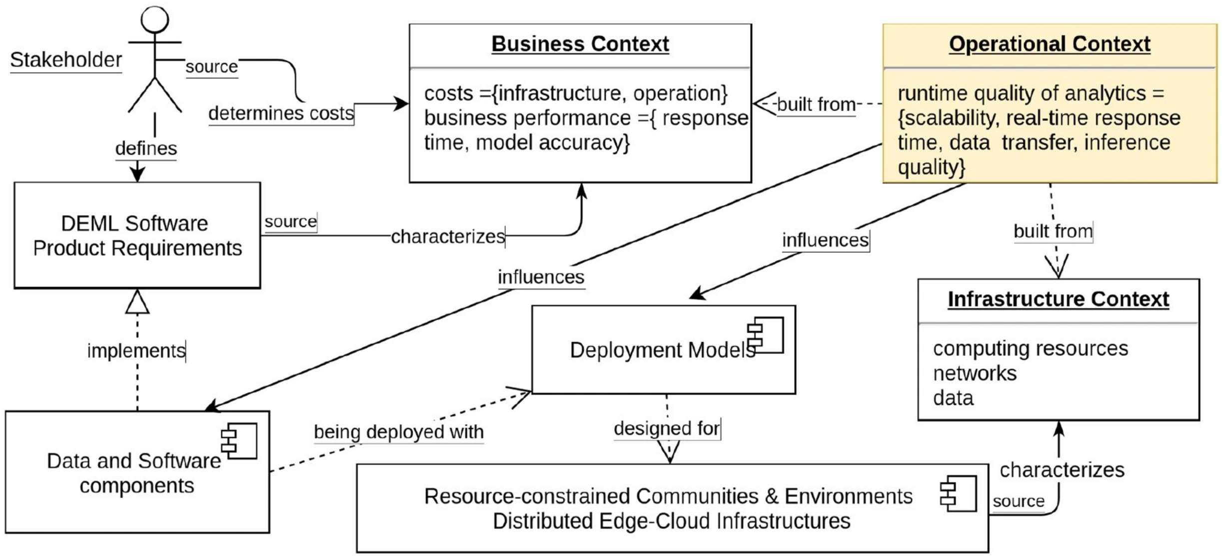
- **Training in cloud and/or on-premise, and inferences in the edge**
  - issues of optimization, loss in transferring/conversion
  - accuracy loss due to the conversion
  - finding suitable edge machines for deployment (e.g., dynamic ML provisioning)
- **Training and inferences in the edge**
  - difficult with tools and resources
  - accuracy loss due to the (limited) training

Check:
https://blogs.gartner.com/paul-debeasi/files/2019/01/Train-versus-Inference.png
https://developer.qualcomm.com/sites/default/files/docs/snpe/overview.html

Aalto University
School of Science

Our focus: to research and practice selected ML engineering analytics topics

Understand contexts for ML systems development and operations?



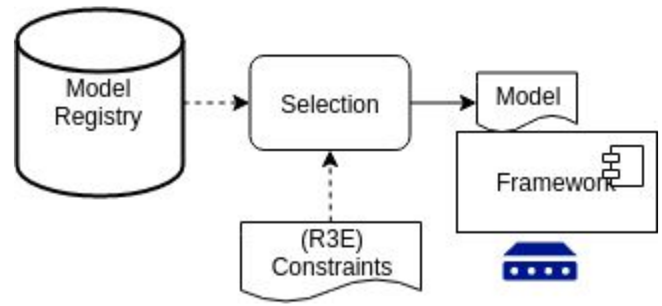
Source: https://link.springer.com/article/10.1007/s40860-022-00176-3

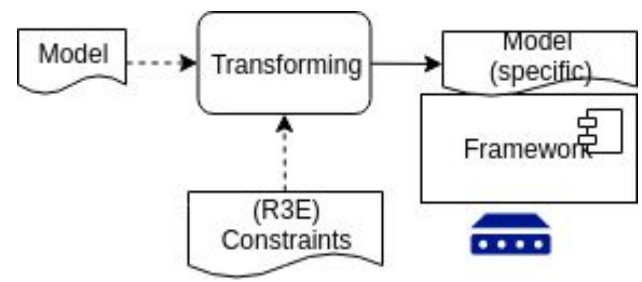
Multiple levels of optimization

Scope/level of abstraction	Research issues
Edge ML Service	ML serving, ML elasticity, ML observability and policies
Cross edge machines ML platforms/infrastructures	ML function partitioning, distributed computation, orchestration, provisioning/(deployment, monitoring ..
In-device/-machine ML platform	Device-machine specific optimization

Area 1: ML model selection and conversion

- **Model management and selection based on R3E**
  - inference accuracy, precision and time tradeoffs with computational requirements
  - work with microcontrollers and accelerators
- **Optimizing/transforming considered R3E requirements**
  - a model can be supported by different frameworks





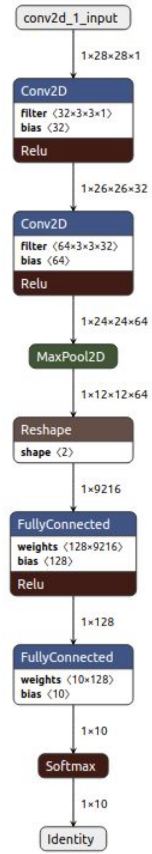
Model optimization

- **Pruning**
  - prune graphs for training, remove insignificant features in ML models
- **Quantization**
  - reduce precision representation → memory storage, or memory bandwidth, power consumption
- **Conditional computation/Regularization**
  - activate certain units of the model
- **CPU/GPU/Accelerator partitioning**

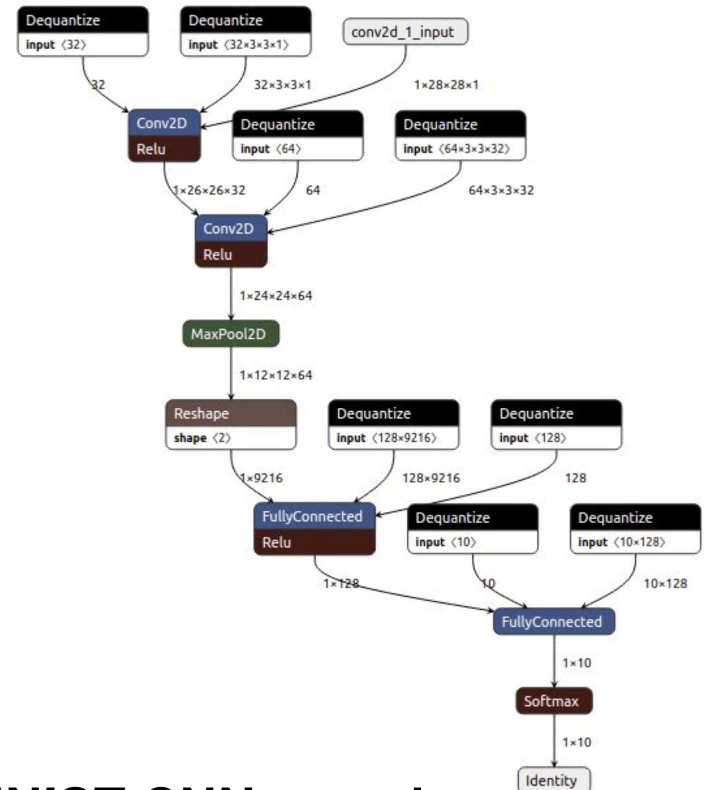
→ **How to consider R3E attributes?**

Example of quantification by reducing floating point

32 bit floating point



16 bit floating point



MNIST CNN sample

Tools/frameworks “the ML compiler/optimizer”

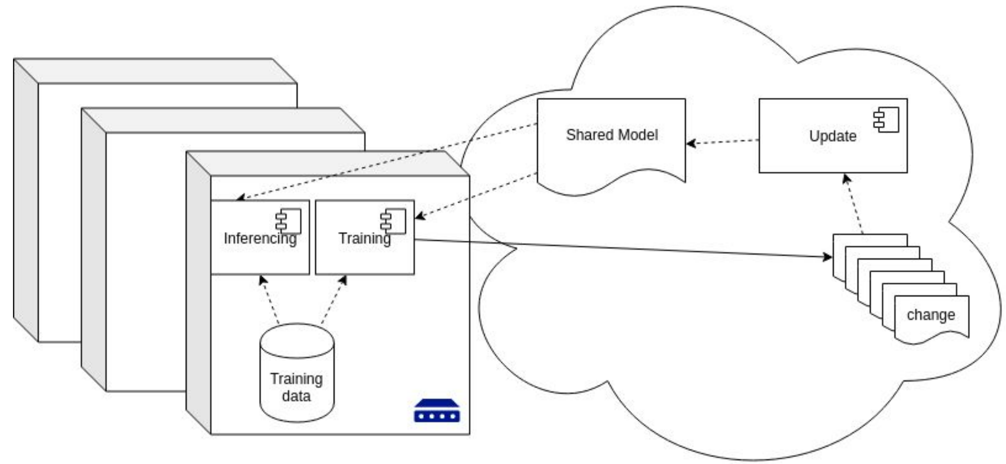
- **ONNX (Open Neural Network Exchange, https://onnx.ai/) format**
  - can be used as an intermediate representation compiled/optimized by tools to specific targets (e.g., https://github.com/onnx/optimizer)
- **Nvidia TensorRT**
  - JetPack SDK (https://developer.nvidia.com/embedded/jetpack)
- **OpenVINO (https://docs.openvinotoolkit.org/latest/index.html)**
- **Apache TVM (https://tvm.apache.org/)**
- **Glow: https://github.com/pytorch/glow**
- **NNI: https://github.com/microsoft/nni/**
- **Amazon SageMaker Neo https://aws.amazon.com/sagemaker/neo/**

Area 2: distributed ML in the edge

- **Goal:**
  - training: creating models using distributed resources (data + computing)
  - serving: split the inference into edge/cloud compute nodes
- **Training:**
  - data parallelism and model parallelism training/learning
  - federating training
- **Serving:**
  - distributed ML networks: distribute the model graph across edge/cloud systems
  - ensembles: similar and different models **used together** for inference

Federated/distributed training with edges

Decentralized with a distributed set of devices holding data and carrying out (sub) training/inferencing:



Benefit factors: data availability, privacy preservation, performance

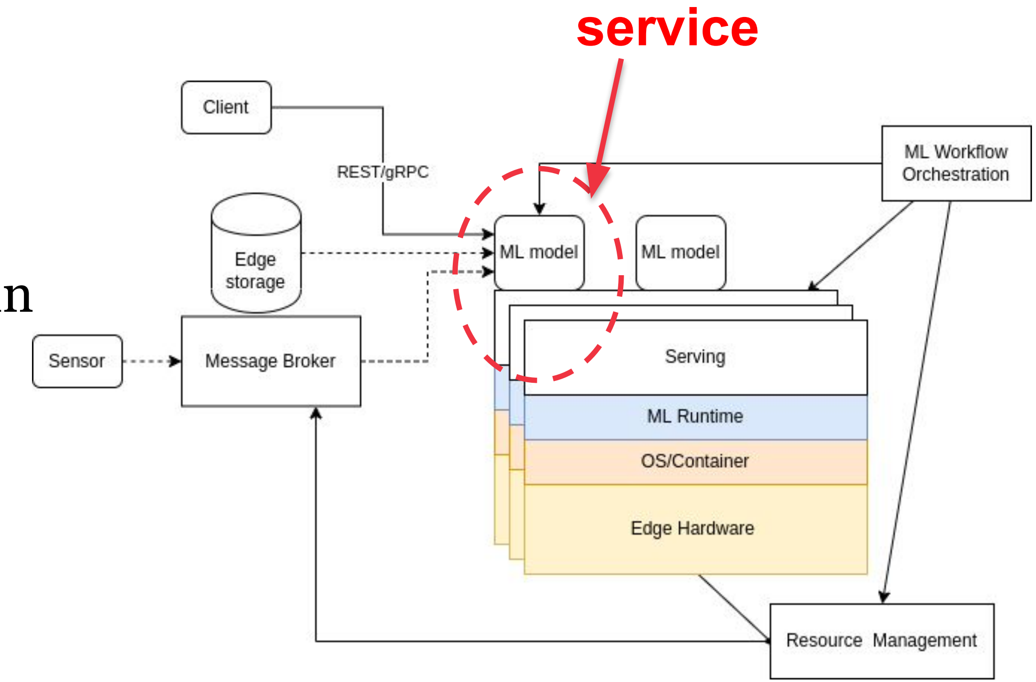
- **Challenges: performance, network, availability (due to sleeping, energy-saving) and trust**
- **What about R3E?**
  - consensus in updates, secured aggregation protocols, reliability and elasticity, and cost

Some tools (not all for edge)

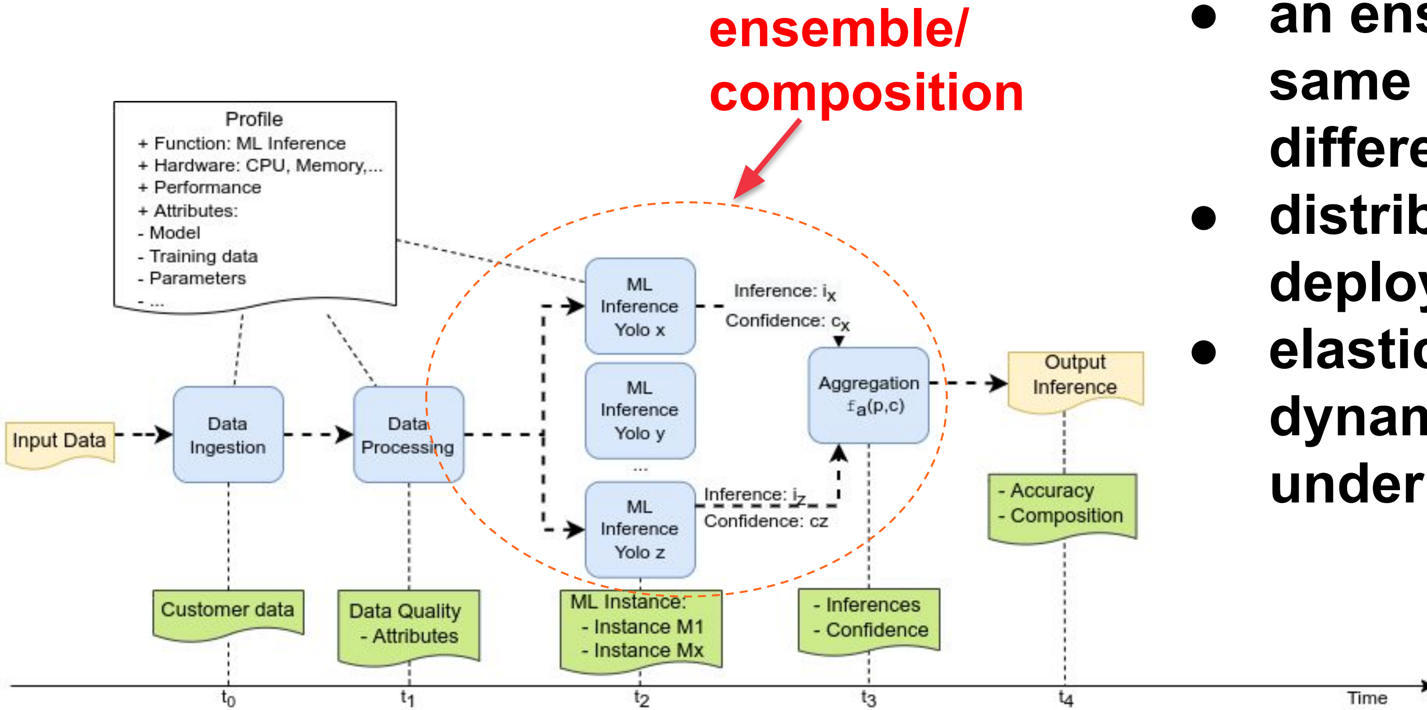
- **Some tools**
  - https://github.com/IBM/federated-learning-lib
  - https://github.com/FederatedAI/FATE
  - https://github.com/tensorflow/federated
  - https://github.com/OpenMined/PySyft
  - https://github.com/horovod/horovod
  - https://flower.dev/
- **Key issues**
  - communications and task distributions
  - resource management, cost and data governance

Parallel and distributed ML serving

- **ML Serving (and R3E)**
  - how to distribute tasks in model serving?
  - how to partition ML tasks in both edge and cloud?
  - how to deal with dynamic reconfigurations of models and ensemble of services



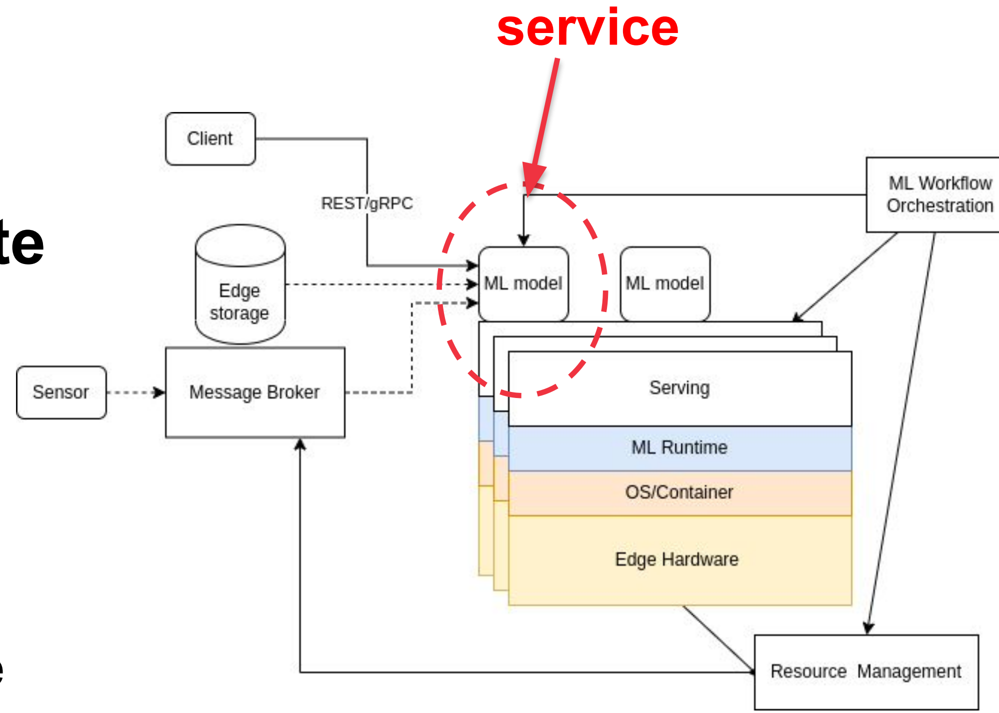
Parallel and distributed ML serving: ensembles



- an ensemble of the same models or different models
- distributed node deployment or not
- elasticity and dynamic control under R3E

Area 3: deployment and update management

- **Secure deployment**
  - model and service
- **Model and software update**
  - not just about ML models
  - runtime update of ML models & services
- **Monitoring of change behaviors**
  - software packages and runtime behavior for update management



Area 4: (Near) real-time ML systems

- **Inferences at near real-time vs continual training**
  - how to make everything works in near real-time?
- **Inferences:**
  - inference serving is integrated into real time data pipeline and decision making
    - *ML serving design would be different due to performance and hosting environment, concurrency capacities, etc.*
- **Training (continual learning) ⇒ continuous ML (runtime adaptation)**
  - data flows to continual training in near real-time ⇒ features/labels are updated real-time for training ⇒ data transformation, feature extraction ⇒ (new, better) model management and serving

Study log

- <u>No study log but </u>read papers and do the hands-on tutorial
- **You can select some issues mentioned as the topic for your individual project**
  - Or incorporate some ideas into your individual project

- **ML with edge systems will increasingly be developed for many advanced software systems!**
  - Good areas for master theses/research projects.

Thanks!

Hong-Linh Truong
Department of Computer Science

rdsea.github.io